***DETAILED ACTION***

1. Applicant's amendments dated May 4, 2009; June 22, 2009; and July 31, 2009 respectively responding to the Office action mailed February 4, 2009 provided in the rejection of claims 1-13, wherein claims 1 and 6 have been amended; claim 5 has been canceled; and claims 23-31 have been newly added.

***EXAMINER'S AMENDMENT***

2. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

3. Authorization for this examiner's amendment was given in a telephone interview by Mr. Gero G. McClellan (Reg. No. 44,227) on July 14, 2009 to obviate any potential 35 U.S.C 112, second paragraph issues and to place the claims in the condition for allowance.

4. In the interest of compact prosecution, the examiner is authorized to further amend the claims 1-4, 6-8, 23-25, and 31 (see Examiner's Amendment below) and to obviate any potential 35 U.S.C 112, second paragraph issues.

5. The application has been amended as follows:

**IN THE CLAIMS,**

Please amend the claims as follows:

1. (Currently Amended) A computer-implemented method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

~~providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator; and wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator and the specified memory allocator of the respective relationship;~~

providing a computer user interface;

allowing a user to establish, via the computer user interface, a relationship between a memory deallocator and a memory allocator, ~~the relationship being stored in the provided data structure~~ wherein the user-established relationship requires that memory space allocated by the memory allocator is freed by the memory deallocator;

allowing the code to execute;

upon a call to ~~a~~ the memory deallocator to free a memory space, determining, by operation of one or more computer processors, whether the call violates the user-established relationship, wherein determining whether the call violates the user-established relationship comprises, for the memory deallocator called to free the memory space:

determining an associated memory allocator responsible for allocating the memory space; ~~and~~

providing an internal relationship, represented in memory, between the called memory deallocator and the associated memory allocator; and

determining whether the internal relationship violates the user-established relationship ~~specifies only one of the associated allocator and the called deallocator~~; and

upon determining that the call violates the user-established relationship, ~~notifying the user~~ storing, in a data structure, a reference to the called memory deallocator and a reference to the associated memory allocator, whereby the internal relationship is recorded.

2. (Currently Amended) The method of claim 1, further comprising notifying the user upon determining that the call violates the user-established relationship, wherein notifying the user comprises halting execution of the code.

3. (Currently Amended) The method of claim ~~1~~ 2, wherein notifying the user further comprises ~~halting execution of the code and~~ displaying a status message to the user.

4. (Currently Amended) The method of claim 1, ~~if the user-established relationship is not violated,~~ further comprising:

upon determining that the call does not violate the user-established relationship, freeing the memory space.

5. (Cancelled)

6. (Currently Amended) A computer-implemented method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

~~providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator; and wherein the data structure stores, for each relationship, a reference to both the~~

~~specified memory deallocator and the specified memory allocator of the respective relationship;~~

establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, ~~the relationship being stored in the provided data structure~~ wherein the user-established relationship requires that memory space freed by the user-selected memory deallocator has been allocated by the user-selected memory allocator;

allowing the code to execute;

upon a call to the user-selected memory deallocator to free a memory space, determining, by operation of one or more computer processors, whether the ~~memory space was allocated by the user-selected memory allocator~~ call violates the established relationship, wherein determining whether the call violates the established relationship comprises, for the user-selected memory deallocator called to free the memory space:

determining an associated memory allocator responsible for allocating the memory space;

providing an internal relationship, represented in memory, between the called memory deallocator and the associated memory allocator; and

determining whether the internal relationship violates the established relationship; and

~~if not, notifying the user that the relationship is violated~~ upon determining that the call violates the established relationship, storing, in a data structure, a reference to the

called memory deallocator and a reference to the associated memory allocator, whereby the internal relationship is recorded.

7. (Currently Amended) The method of claim 6, further comprising notifying the user upon determining that the call violates the established relationship, and wherein notifying the user comprises halting execution of the code and displaying a status message to the user.

8. (Currently Amended) A method for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, comprising:

~~providing a data structure configured to record upper limits on the amount of memory space that the memory allocators can allocate during execution of the code, each upper limit being specific to one of the memory allocators, wherein the data structure stores, for each memory allocator, both: (i) a reference to the respective memory allocator and (ii) the upper limit for the respective memory allocator;~~

setting an upper limit ~~for~~ on the amount of memory space a memory allocator can allocate during execution of the code, wherein the upper limit is specific to the memory allocator, and whereby a relationship between the upper limit and the memory allocator is established~~, wherein both the upper limit and a reference to the memory allocator are stored in the provided data structure~~;

during execution of the code, tracking, by operation of one or more computer processors, the amount of memory space allocated by the memory allocator; ~~and~~

providing an internal relationship, represented in memory, between the memory allocator and the tracked amount of memory space; and

~~when the amount of memory space allocated exceeds the upper limit, notifying a user~~ upon determining that the internal relationship violates the established relationship, storing, in a data structure, a reference to the memory allocator and a reference to the tracked amount of memory space, whereby the internal relationship is recorded.

9. (Currently Amended) The method of claim 8, wherein the internal relationship violates the established relationship when the amount of memory space allocated exceeds the upper limit, and wherein the step of tracking comprises:

determining whether the memory allocator is called to allocate memory and, if so, incrementing a counter; and

determining whether a memory deallocator is called to deallocate memory allocated by the memory allocator and, if so, decrementing the counter.

10. (Previously Presented) The method of claim 8, wherein the step of tracking comprises incrementing a counter in the event of memory allocation by the memory allocator and decrementing the counter in the event of memory deallocation of memory space allocated by the memory allocator.

11. (Currently Amended) The method of claim 8, further comprising notifying the user when the internal relationship violates the established relationship, wherein notifying the user comprises halting execution of the code.

12. (Original) The method of claim 8, wherein the upper limit is independent of other memory size limitations.

13. (Original) The method of claim 8, wherein the upper limit is not a limit on a stack size.

14-22. (Canceled)

23. (Currently Amended) A computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

~~providing a data structure configured to record relationships between the memory deallocators and the memory allocators, wherein each of the relationships specify a memory allocator and a memory deallocator, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator is freed by the specified memory deallocator; and (ii) that memory space freed by the specified memory deallocator was allocated by the specified memory allocator;~~

~~and wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator and the specified memory allocator of the respective relationship;~~

establishing a relationship between a user-selected memory deallocator and a user-selected memory allocator, wherein the relationship requires that memory space freed by the user-selected memory deallocator has been allocated by the user-selected memory allocator;

allowing the code to execute;

upon a call to the user-selected memory deallocator to free a memory space, determining, by operation of one or more computer processors, whether the ~~memory space was allocated by the user-selected memory allocator~~ call violates the established relationship, wherein determining whether the call violates the established relationship comprises, for the user-selected memory deallocator called to free the memory space:

determining an associated memory allocator responsible for allocating the memory space;

providing an internal relationship, represented in memory, between the called memory deallocator and the associated memory allocator; and

determining whether the internal relationship violates the established relationship; and

~~if not, notifying the user that the relationship is violated~~ upon determining that the call violates the established relationship, storing, in a data structure, a reference to the

called memory deallocator and a reference to the associated memory allocator, whereby the internal relationship is recorded.

24. (Currently Amended) The computer readable storage medium of claim 23, further comprising notifying the user upon determining that the call violates the established relationship, and wherein notifying the user comprises halting execution of the code and displaying a status message to the user.

25. (Currently Amended) A computer readable storage medium containing a program which, when executed, performs an operation for managing memory available for dynamic allocation during execution of code containing a plurality of memory allocators and a plurality of memory deallocators, the operation comprising:

~~providing a data structure configured to record upper limits on the amount of memory space that the memory allocators can allocate during execution of the code, each upper limit being specific to one of the memory allocators, wherein the data structure stores, for each memory allocator, both: (i) a reference to the respective memory allocator and (ii) the upper limit for the respective memory allocator;~~

setting an upper limit ~~for~~ on the amount of memory space a memory allocator can allocate during execution of the code, wherein the upper limit is specific to the memory allocator, and whereby a relationship between the upper limit and the memory allocator is established~~, wherein both the upper limit and a reference to the memory allocator are stored in the provided data structure~~;

during execution of the code, tracking, by operation of one or more computer processors, the amount of memory space allocated by the memory allocator; ~~and~~

providing an internal relationship, represented in memory, between the memory allocator and the tracked amount of memory space; and

~~when the amount of memory space allocated exceeds the upper limit, notifying a user~~ upon determining that the internal relationship violates the established relationship, storing, in a data structure, a reference to the memory allocator and a reference to the tracked amount of memory space, whereby the internal relationship is recorded.

26. (Currently Amended) The computer readable storage medium of claim 25, wherein the internal relationship violates the established relationship when the amount of memory space allocated exceeds the upper limit, and wherein the step of tracking comprises:

determining whether the memory allocator is called to allocate memory and, if so, incrementing a counter; and

determining whether a memory deallocator is called to deallocate memory allocated by the memory allocator and, if so, decrementing the counter.

27. (Previously Presented) The computer readable storage medium of claim 25, wherein the step of tracking comprises incrementing a counter in the event of memory allocation by the memory allocator and decrementing the counter in the event of memory deallocation of memory space allocated by the memory allocator.

28. (Currently Amended) The computer readable storage medium of claim 25, further comprising notifying the user when the internal relationship violates the established relationship, wherein notifying the user comprises halting execution of the code.

29. (Previously Presented) The computer readable storage medium of claim 25, wherein the upper limit is independent of other memory size limitations.

30. (Previously Presented) The computer readable storage medium of claim 25, wherein the upper limit is not a limit on a stack size.

31. (Currently Amended) A computer system comprising: an output device, a memory device, one or more computer processors, code resident in the memory device and containing a plurality of memory allocator calls and a plurality of memory deallocator calls, a heap manager resident in the memory device to allocate and free memory of the memory device, and a debugger program resident in the memory device, ~~and a data structure resident in the memory device and configured to record relationships between the memory deallocator calls and the memory allocator calls, wherein each of the relationships specify a memory allocator call and a memory deallocator call, and wherein each of the relationships requires at least one of: (i) that memory space allocated by the specified memory allocator call is freed by the specified~~

~~memory deallocator call; and (ii) that memory space freed by the specified memory deallocator call was allocated by the specified memory allocator call; wherein the data structure stores, for each relationship, a reference to both the specified memory deallocator call and the specified memory allocator call of the respective relationship,~~ and wherein the debugger program ~~comprising~~ comprises a debugger user interface configured to at least:

allow a user to view allocation/deallocation history information at a user-specified memory location; and

allow a user to establish a relationship between a memory deallocator call and a memory allocator call~~, wherein the relationship is stored in the data structure, and~~; wherein the user-established relationship requires that memory space allocated by the memory allocator call is freed by the memory deallocator call; wherein an associated memory allocator call is determined for the memory deallocator call; wherein an internal relationship, represented in memory, between the memory deallocator call and the associated memory allocator call is provided; wherein a violation of the a requirement of the user-established relationship causes the debugger user interface to ~~notify the user~~ store, in a data structure, a reference to the memory deallocator call and a reference to the associated memory allocator call, whereby the internal relationship is recorded; and wherein the requirement is violated when the internal relationship violates the user-established relationship.

- **END OF AMENDMENT –**

***Allowable Subject Matter***

6. Claims 1-4, 6-13, and 23-31 (renumbered as 1-21) are allowed.

7. The following is an examiner's statement of reasons for allowance:

The cited prior art taken alone or in combination fails to suggest

"... providing an internal relationship, represented in memory, between the called memory deallocator and the associated memory allocator; and

determining whether the internal relationship violates the user-established relationship

... storing, in a data structure, a reference to the called memory deallocator and a reference to the associated memory allocator, whereby the internal relationship is recorded", as recited in independent claims 1 and similarly recited in independent claims 6, 8, 23, 25, and 31.

8. Claims (2-4), (7), (9-13), (24) and (26-30) are considered allowable by virtue of their dependence on allowable independent claims 1, 6, 8, 23, and 25 respectively.

9. Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance.

***Conclusion***

10. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is (571) 270-1240. The examiner can normally be reached on 8:00-5:30 (EST/EDT), Monday through Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Ben C. Wang
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192